OpenBiblio Documentation

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Introduction

This manual provides documentation for users of OpenBiblio 1.0, an automated library system.

OpenBiblio 1.0 has a public OPAC interface and a staff interface. You can successfully use either interface on your computer, tablet, or phone.

About This Documentation

This documentation uses ASCIIDoc, using a workflow borrowed from the Evergreen ILS Documentation Interest Group.

The OpenBiblio Community

We are happy to welcome you to the OpenBiblio Community! You have joined a worldwide group of librarians, developers, community centers.

Getting help

You can get help from the mailing list or the sourceforge forums.

Reporting bugs

You can report bugs in the bitbucket repository.

Creating plugins

Instructions coming soon.

Installing OpenBiblio

System requirements

We have tested OpenBiblio installation on Windows 7 and Ubuntu Linux 14.04.

We also plan to test other versions of Windows, Ubuntu Linux 12.04, Debian Linux, Fedora Linux, Mac OSX, and CentOS Linux.

To install OpenBiblio, you will need PHP version 5.4 or higher. We have not yet tested OpenBiblio on PHP 6.x or 7.x.

You will need MySQL. Not sure of version.

You will also need Apache.

Installing OpenBiblio 1.0 on a Windows computer

You can turn your personal Windows computer into an OpenBiblio server. You will need administrator access to your computer, an Internet connection, and you may also need to check with your local IT experts that you are not behind a restrictive firewall.

Note that turning your personal computer into a server carries some risks, especially if your server is broadcasting across the web instead of just a local network. We recommend that you run this software on a dedicated computer that does not contain any sensitive or private data.

Install prerequisite software

1. Download EasyPHP's Webserver from http://easyphp.org. We have tested these instructions with version 14.1 of EasyPHP.
2. Agree to the relevant licenses and choose an installation folder that is convenient for you.
3. Once EasyPHP is installed, it will appear as an icon in your system tray. On Windows 7, click on this icon and open the dashboard.
4. The dashboard will open in your default Web browser. Once it does, click Settings in the top right corner of the screen.
5. Click on HTTP SERVER. Install and start this service.
6. Click on DB SERVER. Install and start this service.
7. Go back to the dashboard.
8. Under modules, you should now see an option to administer MySQL through a program called PhpMyAdmin.
9. Open PhpMyAdmin. The default username is root and there is no default password. Click on Users.
10. Click on "Edit Privileges" next to root.
11. Click on "Change Password".
12. Click on Add user.
13. Choose a memorable username, like obiblio, and enter it. In the host field, enter "localhost". Create a long, secure, memorable password.
14. Under "Database for user", there should be a checkbox for "Create database with same name and grant all privileges". Check this box.
15. Click go.

Install OpenBiblio

1. Open the EasyPHP dashboard.
2. Go to Settings and open the HTTP SERVER settings.
3. Note the Document Root field. This is the folder on your computer where OpenBiblio will be running.
4. Open the DB SERVER settings. Note the parameters area.
5. Download OpenBiblio and place the files into the Document Root folder.
6. Open the document root folder.
7. Locate the file database_constants_deploy.php. Make a copy of this file called database_constants.php.
8. Edit your new database_constants.php file. Enter the database, username, and password you created in PhpMyAdmin. The hostname should be the IP address and port that you noticed in the DB SERVER settings in the format 192.168.333.333:3388, where 192.168.333.333 is the IP address and 3388 is port number.
9. Save the file.
10. Go back to the EasyPHP dashboard and open the HTTP SERVER settings once more.
11. There is a link there to open up OpenBiblio under the URL field.
12. Add /install to the end of the URL in your browser. The installer should run.

Using the OpenBiblio installer.

1. Install test data if you would like.
2. Log in to your new OpenBiblio system. The default user is admin with password admin.

Installing OpenBiblio 1.0 on a shared Linux host

These instructions will help you set up OpenBiblio on a Web server provided by Dreamhost, Hostgator, or a similar service.

1. Create a database using your hosting service's interface. Note the host, username, password, and database name.
2. Download the Filezilla FTP program

Installing OpenBiblio 1.0 on a Linux host with full access

These instructions consist mainly of commands that you can run in your terminal to quickly get a working installation of OpenBiblio. If you are not comfortable with using the terminal, feel free to use the instructions above for a shared Linux host.

These procedures have been tested on Fedora 21 and Ubuntu 14.04.

Install prerequisite software

OpenBiblio requires apache, PHP, and MySQL. If you intend to copy catalog records from other libraries, we strongly recommend that you use yaz. The following steps include a YAZ install.

On Ubuntu:

```
sudo apt-get install apache2 php5 mysql-server yaz php5-dev php5-mysql php-pear libyaz-4  ↩
    unzip
```

Tip
This will prompt you to choose a root password for MySQL. Make sure it is a secure one!

On Fedora 21:

```
sudo yum install httpd php mysql-server php-devel php-mysql php-pear unzip yaz libyaz-devel ↩
     gcc
sudo mysql_secure_installation
```

Tip
mysql_secure_installation will ask you for a current root password, which you may leave blank. Be sure to give root a new, secure password. You should also choose to remove anonymous users, disallow root login remotely, and remove the test database.

Install yaz (optional)

Next, install yaz as a PHP extension using pecl.

```
sudo pecl install yaz
```

Add a new line to your php.ini file. This file can be found at /etc/php/php.ini in Fedora and /etc/php5/apache/php/ini in Ubuntu. The line should be something like the following:

```
extension=yaz.so
```

Start the Apache and MySQL services

On Fedora 21:

```
sudo service httpd start
sudo service mysqld start
```

On Ubuntu:

```
sudo service apache2 start
sudo service mysql start
```

Create a MySQL database

First, open up MySQL.

```
mysql -uroot -p #MySQL will ask you for the root password you just created
```

Next, type the following commands to create a user and database for OpenBiblio to use.

```
CREATE DATABASE obiblio;
GRANT ALL PRIVILEGES ON obiblio.* TO obib_user@localhost IDENTIFIED BY 'obib_password';
```

Prepare OpenBiblio for its installation

```
wget -O obib.zip https://bitbucket.org/mstetson/obiblio-1.0-wip/get/26f88507b580.zip
unzip obib.zip
cd new-obib-dir
cp database_constants_deploy.php database_constants.php
```

Edit database_constants.php with the text editor of your choice.

Delete the default "It Works" page from /var/www/html.

```
sudo mv ./* /var/www/html/
```

On Fedora 21, you may need to run the following command to configure SeLinux to allow access to the folder.

```
restorecon -r /var/www/html
```

Using the OpenBiblio installer.

1. Install test data if you would like.
2. Log in to your new OpenBiblio system. The default user is admin with password admin.

OpenBiblio System Administration

Thanks for being an OpenBiblio system administrator. We're glad that you are helping your library in this way.

Managing your libraries

OpenBiblio 1.0 introduced the ability to manage multiple libraries with a single OpenBiblio installation. Each library is considered a separate "site," with its own name, collection, address, hours, closed dates, and other settings.

Adding a site

1. Go to Admin → Sites.
2. Click "Add New".
3. Fill out the relevant information. Choose the calendar that includes the dates your new library will be closed. See below for more information about the calendars.

Warning
If you do not choose a calendar, the new site will not be able to calculate due dates properly and you will not be able to check out any copies at this site.

4. Click "Add."

Deleting a site

1. Ensure that there are no copies at the site that you'd like to delete.
2. Sign in as a different site.
3. Go to Admin → Sites.
4. Delete the site.

Warning
Don't delete a site when you are logged in, or you will cause a permanent error state that won't be resolved until you manually add the site back into the database.

Warning
If you delete a site that has copies attached, they will show as "Undefined" in the system.

Managing your users

To add a new staff member, go to Admin → Staff Admin. Click on the "Add New" button, found both at the top and bottom of this screen. You can then assign a username, password, name, and permission set to your staff member.

Warning
You must assign at least one permission, or the staff member will not be able to log in.

Working with passwords

Openbiblio's password management system is very straightforward. Users with Admin permission are able to change their own password or the password of any other staff account (including other administrators). Users without the Admin permission may not change anybody's password, not even their own.

To change the password of a staff account, go to Admin → Staff Admin. Click the pwd button next to the account you are working with. Enter the new password.

Managing your catalog

This section describes the following settings in the Admin menu.

Managing media types

Media types link

Managing collections

Collections

Managing displayed fields

Biblio fields

Managing custom fields for copies

Biblio copy fields

Managing copy catalog options

Z39.50 and SRU

Configuring the OPAC

This section will describe settings found in the admin menu.

Managing locales

OpenBiblio can be translated into any language that you wish!

Character set

You can use any character set that you find by running the MySQL command SHOW CHARACTER SET. Do this by going to Admin → Locale and entering your preferred character set. If you enter an invalid charset or leave this field blank, it will default to the *utf8* character set.

Note
The character set you specify will appear in the `<meta charset>` attribute and apply to any database calls.

Creating a new locale

This requires access to the files and a good UTF-8 text editor.

Create a directory for your new locale

1. Create a new directory in the locale directory with the 2-character ISO 639-1 code for the language. For example, if you would like to do an Amharic translation, your directory would be called `am`.
2. Create a PHP file called metadata.php. Make sure that the class name begins with the ISO 639-1 code for your language.

```
<?php
class amMetaData {
        public function __construct() {
                $this->locale_description = "Amharic";
        }
        function pluralForm($n) {
                if ($n == 1 or $n == -1) {
                        return 'singular';
                } else {
                        return 'plural';
                }
        }
        function moneyFormat($amount) {
                if ($amount < 0) {
                        return sprintf("(-%.br)", abs($amount));
                } else {
                        return sprintf("%.br", $amount);
                }
        }
}
```

3. Add translations to trans.php.

Managing the MySQL database

This section will describe the database

Managing the server

This section will discuss other server-level considerations.

Logs

This section will mention logs of particular interest to OpenBiblio administrators.

Circulation

The circulation module is the heart of the OpenBiblio system.

Checking out materials

To check out materials, you must first open a member's account.

Retrieving a member's account.

1. Click on Circulation on the left side of the screen.
2. Enter a portion of the member's last name. If you have member barcodes turned on, you may alternatively scan their barcode.

Tip
Anybody with administrator privileges can turn member barcodes on using Tools → System Settings → Use Member Barcodes.

Checking out an item

1. Type or scan the item barcode into the Check Out field.
2. Any items you have checked out will appear on the screen, provided there are no errors in the patron or copy records.
3. If there are errors (often a mis-scanned barcode), they will appear at the bottom of the screen.

Checking in materials

1. Click on Circulation on the left side of the screen.
2. Click Check In.
3. Type or scan the barcode number.
4. If the book was checked out, it will be added to the shelving cart, which is represented by a list at the bottom of the screen.

What if it is a weird status, does it change to regular?

How do you empty the shelving cart?

What if there is a hold or fines on the item?

Showing a member's circulation history

OpenBiblio stores information about copies that a member has checked out previously. You can access this list after you retrieve a member account.

Retrieving a member's account.

1. Click on Circulation on the left side of the screen.
2. Enter a portion of the member's last name. If you have member barcodes turned on, you may alternatively scan their barcode.

Viewing the history

Click on the History button, near the top of the screen.

Note
Unlike some ILSs, OpenBiblio does not anonymize circulations after a certain period of time has elapsed. Please know that this can be considered a member confidentiality issue in some jurisdictions.

Holds

You can place holds on items so you can check them out at a later time.

Bookings

You can also book certain items

Cataloging

OpenBiblio offers several ways to get your materials into your catalog and to manage the records you have entered.

Orientation to OpenBiblio cataloging

OpenBiblio uses the MARC format as the basis for the catalog, but its editing interfaces display human-readable field labels, rather than specific numeric MARC tags.

OpenBiblio's use of MARC is pretty standard, but there are a few specific ways it treats certain fields.

Call numbers

OpenBiblio's Call Number searches search only 099$a, which is the local call number field. However, if you'd like to add call numbers from other fields, you can enable the Call Number Utilities plugin, which allows you to populate the 099$a fields with data from other call number fields to make them searchable. Instructions for using this plugin can be found in the OpenBiblio Plugin manual.

Special subfields

Three subfields display somewhat differently than usual.

- Field 024 $a (identifier) is assumed to be a DOI, so it displays as a link to that digital object.
- Field 505 $a (contents) is typically very long, so it displays as a scrollable text field.
- Field 856 $u (URL - included by default in the ebook and web site material types) displays as a hyperlink.

Copy cataloging

1. Click Cataloging.
2. Click New Item.
3. Type or scan an ISBN. If you are unable to find any results using an ISBN, you can use the dropdown menus to try other search criteria.
4. If it finds a catalog record from a different library, it will fill in all the fields that it can, based on the media type and information from the other library.
5. Make sure that the call number is correct.

   If you are seeing the wrong type of call number (e.g. you see a LoC call number when your library uses Dewey Decimal), somebody with admin privileges will need to change the Admin → Online Options settings.
6. Click Submit.

Adding an item

After you add your record, you will be able to attach an actual, physical item to it.

If your library is using barcodes, you will be able to scan or type your barcode into the barcode number field. If you have the Auto Barcode option selected, OpenBiblio will generate a barcode for you. If you are not using barcodes, OpenBiblio will generate one that will only be used for internal purposes.

If item Photos are enabled, the user will be given to capture an image of the item (usually the dust cover). This feature requires a webcam or equiv. Enabling this feature will also cause the user to be ocassionally *bugged* to formally allow the browser to take photos - this is a HTML5 *feature* designed to ensure user privacy. Once photos are available, they will show up on various reports, biblio lists, etc.

Original cataloging

You may not be able to find a record from another library. To do so, go to Cataloging → New Item and click the *Manual Entry* button.

1. Select the appropriate media type. The form will then show the fields appropriate to that media type.

   Note
   Anybody with the Admin permission may add, remove, or modify which fields are shown in this screen by going to Admin → Biblio Fields.

2. Be sure to fill in any fields marked with the * character. These fields must be filled in before you finish your record.
3. If you would like to edit your record further before displaying it to your patrons, unselect the *Show in OPAC* checkbox. With this unselected, members will not be able to view your record. You may change this setting at a later time.

Adding an item

After you add your record, you will be able to attach an actual, physical item to it.

If your library is using barcodes, you will be able to scan or type your barcode into the barcode number field. If you have the Auto Barcode option selected, OpenBiblio will generate a barcode for you. If you are not using barcodes, OpenBiblio will generate one that will only be used for internal purposes.

If item Photos are enabled, the user will be given to capture an image of the item (usually the dust cover). This feature requires a webcam or equiv. Enabling this feature will also cause the user to be ocassionally *bugged* to formally allow the browser to take photos - this is a HTML5 *feature* designed to ensure user privacy. Once photos are available, they will show up on various reports, biblio lists, etc.

Importing MARC records

By default, the MARC record importer imports items as a "test load," so you can verify that there are no errors. You can set "test load" to false once you are satisfied with your settings and file.

Tip
MARC records can use any one of a number of different encoding schemes. Before you import any MARC records, be sure that your administrator has selected the correct character set in Admin → Character Set.

Importing CSV records

The CSV File

CSV files are laid out in columns and rows. The first row must contain the heading information.

Some common headings are:

Column header	Meaning
barCo	Barcode Number
media	Media Type
coll	Collection
099$a	Call Number
100$a	Author
245$a	Title
245$h	Medium
306$a	Playing Time
260$c	Date of Publication, Distribution, etc.

To find more headings click on the "Admin" link on the left side of the page. From there click on the link that says "Biblio Fields". There you will find different options to use for headers. The data will go from the second row down in the CSV file. The data needs to be input in the same order as the column headings in the first row. The Media Type heading requires specific data to be input in the column. The data options available for the media column are:

- audio tapes
- book
- cd audio
- cd computer
- equipment
- magazines
- maps
- video/dvd

Importing the CSV file

Click on the "Browse" button, from there you should be prompted to open your CSV file. In the options box you will need to find the "Show all records" drop down. This drop down needs to be set as yes or you will be unable to view the records.

Bulk delete records

This powerful tool will delete any copy barcodes you specify from the database. You may also choose to delete the bibliographic data relevant to a parent item once you have deleted the last copy on an item.

Warning
This action cannot be undone.

Research

This menu area is intended for patrons or staff who do not have access to either the cataloging or circulation facilities.

Local search

This menu implements an on-line card-catalog for the library.

DOI search

A DOI (digital object identifier) can refer to an article, journal, digitized archival object, or any other digital object that may help you in your research. If you have one of these identifiers, you can paste it into the DOI search box, and OpenBiblio will open the relevant resource in a new tab for you.

Cover Photos

A viewer of all library material for which photos are available. The format of the pages displayed are set using the Cover Photo tab of Library Settings under the Admin Menu. Clicking the title of the item shown will bring up the usual descriptive information. Nothing may be edited from these pages.

Cart

Provides a list of material awaiting shelving. Again, clicking on an item will bring up the usual descriptive matateriral.

Reports

OpenBiblio has a very flexible report system.

Running a report

Do it!

Exporting to different formats

It seems like PDF isn't working :-(

Adding new reports

You need access to the server via FTP, RDP, or SSH.

Managing OpenBiblio Plugins

OpenBiblio has a very flexible plugin system. This document will describe how you can enable and disable plugins, understand the plugins that are included with OpenBiblio, and design a new plugin.

Enabling plugins

Tip
You will need access to the Tools menu.

---

1. Go to Tools → Plugin Manager.
2. Make sure that Plugins are Allowed.
3. Check the box next to each plugin you are interested in using.

Plugins bundled with OpenBiblio

These plugins all come bundled with a default OpenBiblio installation.

CSS Utilities (cssUtils)

This plug-in will looks for two categories of CSS issues:

1. CSS entries in . . . /styles.css that appear to not be in use
2. CSS classes referenced in OB that do not appear in . . . /styles.css

WARNINGS

No fixes are suggested or made. The user should be VERY careful in removing any that appear to not be in use as some css usage is dynamic via jQuery and is dificult to detect programatically.

Query selectors having multiple entries (e.g. $(*.reqd sup*)) are not currently parsed as multiple entries to be searched.

Pull-Down List Manager (List Manager)

This plug-in is intended to provide a developer with a simple means to view the contents of pull down lists that are used in various locations throughout OB.

No means are provide at this time to modify these lists; use phpMyAdmin or equiv for that purpose.

Media Fields

This plug-in is intended to provide a means for OB users to exchange input form layouts.

Some media forms are infrequently used and a new user may not know just what material should be collected.

Using this plugin, it is possible to create a text file of a layout that can be emailed to another for them to import.

Orphan File finder (orpahnFiles)

This plug-in is intended to find old abandoned files. The concept here is to find files which: . are not referred to via any menu . are not referred to by any HTML mechanism such as Javascript, CSS, Form Actions, Images, etc. . are not referred to by any PHP mechanism such as 'includ'e s or 'require's . are not referenced in Object class inheritances.

Note
there are many legacy files and folders the have an initial letter'x'. These are to be considered *Legacy* works and may be removed from the OB project at the Lead Developer's descretion.

Note
There is an entire . . . /docs folder that has never been integrated into OB ver 1.0. this folder may be removed from the OB project at the Lead Developer's descretion.

Translation Utilities (transUtils)

This plugin is intended as an aid to the creation or maintenance of a locale translation. All T(...) entries in OB .php files are checked. Each available translation is tested seperately.

The following functions are available:

1. Check for Duplicate Entries. Intended to find duplicate entries which may be out of place alphebetically and so un-noticed.
2. Check for Unused Entries. Look for table entries which have been abandoned, or through miss-spelling somewhere, are not currently being used.
3. Check for Needed Entries. Look for T(...) entries which do not have a corresponding entry in the trans.php file.
4. Check for Potential Entries. Look for quoted strings (both normal display, and error) in all files which do not have a T(...) surround. Error messages embedded in the various language processors (PHP, JS, CSS, mySQL, etc) will not be caught.

Call Number Utilities (CallNoUtils)

This section describes the Call Number Utilities plugin.

A pull-down allows a search for:

1. items without an assigned call number
2. proposed call numbers based on a desired schema
3. posting proposed call numbers as needed.

Find Records without call numbers

This function will scan the entire database and return a list of all biblios which do not have a call number assigned.

Dry Run of call number add

This function will propose call numbers based on the preferred source.

Add call numbers to records

This function will post the proposed call numbers.

Creating a new plugin

This requires basic PHP skills and access to the files.

Design the plugin

1. Create a new directory that begins with the word plugin, followed by an underscore, followed by the name of your plugin. In UNIX systems, you might type something like this:

   ```
   mkdir plugin_excitingPlugin
   ```

2. If you need to include custom CSS or JS, add it to a file called custom_head.php in this new directory.
3. Create three PHP files in the directory: a JS file, a Srvr file, and a Forms file.

Create a plugin form

Your users will interact with your plugin using a form, located at `plugin_excitingPlugin/excitingForms.php`.

A basic plugin consists of a require statement to get the necessary methods, some code establishing basic UI, some HTML form markup, and a section to display results and messages. Here's a simple example:

```
<?php
  require_once("../shared/common.php");

  $tab = "tools";
  $nav = "ExcitingMgr";

  require_once(REL(__FILE__, "../shared/logincheck.php"));
  Page::header(array('nav'=>$tab.'/'.$nav, 'title'=>''));

?>
<h1 id="pageHdr" class="title"><?php echo T("excitingMgr"); ?></h1>
    <section>
        <fieldset id="exciting">
            <label for="site_cd" >Choose your favorite library:<select id="site_cd" ></ ↩
                select></label>
            <input type="button" id="showResultsBtn" value="Show results" />
        </fieldset>
    </section>
    <div id="rslts"></div>

<?php
require_once(REL(__FILE__, "excitingJs.php"));
?>
```

Create a plugin JS file

Your plugin's JS file — located at `plugin_excitingPlugin/excitingJs.php` — is responsible for filling your Form with dynamic data, whether it is filling drop-down menus or presenting the results of your query from the server.

A basic JS file consists of functions that capture data from the user and send it to the server file(s). Here's a simple example:

```
<script language="JavaScript" defer>
"use strict";

var excite = {
        init: function () {
                excite.url = 'excitingSrvr.php';
                excite.listSrvr = "../shared/listSrvr.php";
                excite.fetchSiteList();

                $('#showResultsBtn').on('click',null,excite.doSendUserData);
        },
        fetchSiteList: function () {
                $.getJSON(excite.listSrvr,{mode:'getSiteList'}, function(data){
                var html = '';
                for (var n in data) {
                    html+= '<option value="'+n+'">'+data[n]+'</option>';
                }
                $('#site_cd').html(html);
                });
        },
        doSendUserData: function () {
                $.post(excite.url, {'mode':$("#site_cd").val()},
                        function (response) {
                                $('#rslts').html(response);
```

```
                                $('#rslts').show();
                });

        },
};
$(document).ready(excite.init);

</script>
```

Create a plugin server

Your plugin's server — located at `plugin_excitingPlugin/excitingSrvr.php` — is responsible for three tasks:

1. Performing any actions the user requests.
2. Obtaining any relevant data from the database, file structure, or external API.
3. Preparing a nice HTML display of the relevant data, which your JS file will add to the Form to display to your user.

A basic plugin server consists of a require statement to get the necessary data and methods and a switch statement to take care of the `$_REQUEST['mode']` variable. Here's a simple example:

```
<?php
        require_once('../shared/common.php');
        switch ($_REQUEST['mode']){
                case 1:
                        echo 'That\'s my favorite library too!';
                        break;
                case 2:
                        echo 'That library is pretty good.';
                        break;
                default:
                        echo T('invalid mode');
                        break;
        }
```

If you would like to access data from the database, you should require the appropriate model from the model directory. For example, if you'd like to use the Biblios model, you would include this in your plugin_excitingPlugin/excitingSrvr.php:

```
require_once('../model/Biblios.php');
$bib = new Biblios;
```

You can then access all those great methods and data through the `$bib` object.

Add your plugin to the OpenBiblio menu

1. Create a new directory inside the plugins directory with the name of your plugin. In UNIX systems, you might type something like this:

   ```
   mkdir plugins/excitingPlugin
   ```

2. Create a file within this new directory called nav.nav

   ```
   <?php
   Nav::node('tools/excitingPlugin', 'Exciting Plugin', '../plugin_excitingPlugin/ ↩
       excitingForms.php');
   ```

3. If you'd like to use Obib's localization framework but need to localize some more terms, create a file called tran.tran. You can use this to add new terms to your locale.

> **Tip**
> Plugins already have all of the exciting localization ability built in. However, note that you cannot override localized term definitions within tran.tran.

+ WARNING: There is not yet a place to specify localized terms for plugins.

4. Turn on your plugin using Tools → Plugin Manager.

OpenBiblio Recipes

This is a collection of neat tricks and configurations that might be nice for your library.

A successful initial configuration

1. Go to Admin → Library settings. Set the character set to *utf8*.